P=NP

HMKK

PvsNP

Imagine running an eight hundred meters race, there are 2 of us, I am a cheetah, you are a slug. The race starts and it finished. Now, imagine P=X<31sec and NP=X>31sec. Now, you and I represent the method/algorithm, the distance of the race is the problem, the time it takes us to run the race is the speed, and the finish line is the result. Now, imagine I finished the race in 30sec, and you finished the race in 2hours. I would run in P while you are in NP, yes? The question of PvsNP asks: If I can tell, in P=X<31sec, that the slug won the 800meters race, then can the race of 800meters, which runs by a NP=2hours slug, be run by a P=X<30sec racer? The answer is yes! All you need to do is find a P=X<30sec racer but finding such a racer can be difficult. Which brings up another race/problem(let's call this race race1). How does this other problem come up?

Well, let's say you want to know 10+10=X, first you need to know the value of ten, which can be a separate problem from 10+10=X and in this case, a separate problem. 10+10=X assumes you already know the value of 10. Let's say you only know the value of (1), then you would have to add 1×10 which contains 10 problems, or find the value of 2 then add 2×5, which contains 6 problems, etc. The number of steps you take to find the value of X is the number of problems you have to solve before finding X. Or in the example of the race, you have to start looking for a P=X<3sec racer, that is a new problem. Now, back to our thought experiment: you would need a P=X<31sec racer to finish race1 in P. And if you don't have a racer for race1, that would bring up another race/problem(let's call that race race2), which would need a P=X<31sec racer to finish race2 in P. And this can go on and on. But! What about solving all of them at once in P=X<31sec? Do you realize that most people think if P=NP, then all the problems in the world would be solved? Well, no, that's not true. Look at this: let's assume to solve X, you need to solve 10 separate problems or in other words, win 10 races before you can solve/win the problem/800meters race in P=30sec.

Also, we're going to assume that the algorithms/racers we use to run all 11 problems/races, take 30sec each for all races/problems. Well, to solve all 11 problems or run 11 races at once in 30sec will be impossible since (30sec×10)+30sec=330sec. In reality, all 11 problems/races would take 5 hours and 30sec. So, to solve X, it would take 5 hours and 30sec. Now, imagine you need to solve 10000 problems before solving K=X×Y, and each algorithm takes 30sec, it would take you about 208 days before you can solve K=X×Y in P=30sec. Now, imagine to know K=X×Y, you need to first solve 100000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000 problems, and each takes about 100000000sec to solve, how long would it take you to solve it, very long, right? The reason a problem can take so long to solve is that all the little problems inside such problem, that need to be solved, each need to find a solution, and each (can) take a very long time to find. But……Back to our thought experiment:

To get all 11 races/problems run/solved in 30second, you would have to do something like this: $30sec=(X\times10)+Nsec<30sec$, where N is any number smaller than 30, and $(X\times10)$ along with Nsec is under 30sec. You can't calculate all problems at once, you can only solve one at a time because each problem depends on the other's result. To say "P=NP means all our problems are solved" is very wrong. What it says precisely is that all problems (can) be solved in P, that's all. P problems can also be solved in NP, that's why they are equal, P=NP and NP=P. Any NP problem can be solved in a second or less. Sometimes we find ourselves with an algorithm that is, say, tiger=1sec and tell ourselves that such algorithm is fast, yes it is, but if other problems(let's call these problems problems1) depends on such algorithm(solving its own problem), It's always best if we find a faster algorithm, let's make cheetah=1millisec, that way, Problems1 can be solved in a faster time. Making all the algorithms we have faster should be the priority, despite how fast they already are, because it's so we get the slower ones to be solved in a faster time.

I said this many times, this question is not a computer science question, it's not relative, it's universal, the question belongs to every field. Another thing is if knowledge is infinite, then there is no algorithm for infinite knowledge, since there is always new knowledge for you(the student), to learn. Knowledge is not of that, which is in our heads, we are not, and will never be that special, instead, knowledge is the universe, us a part of. It is wrong to believe that we are in the universe, we are not, we are merely a part of the universe, I am merely a [come together] part of the universe. We will never know everything. We can know everything we know we don't know but never everything we don't know we don't know.